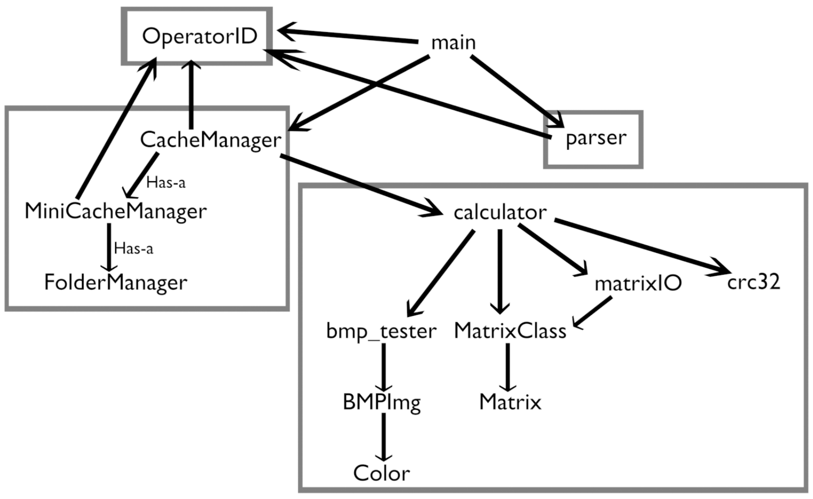
1 Code Hirarchi



2 Why Our Code is SOLID

2.1 S

single responsibility: all parts of our code have a single responsibility. Parser: parses the command OperatorID: contains command's info CacheManager: holds the mini cache managers MiniCacheManager: acts as a cache for each function FolderManager: responsible for operations in each function's folder Calculator: calculates the results for each function MatrixIO: responsible for input and output from files with matrices in themThe rest of our code is from previous exercises.

2.2 O

open-closed principle: Our code can easily be extended to supportmore operations by adding another mini cache manger to the cachemanager and adding a calculating function to the calculator namespace.

2.3 L

Liskov substitution: Our code doesn't use inheritance but we can useour classes as a base for other classes that will work as well.

2.4 I

interface segregation: Each of our classes can function as an interface. If a client wants to change the behaviour of the code he most likely will onlyneed to replace one class with an extension of that class.

2.5 D

dependency inversion: Our code doesn't depend on the classesthemselves as the header files determine the needed functions. classescan implement the functions in different ways but as long as they implement the functions in the header the whole project should work as intended.